Simplified LDAP Access Control Language System

BACKGROUND OF THE INVENTION

TECHNICAL FIELD

The invention relates to accessing information from a directory structure in a computer environment. More particularly, the invention relates to controlling access to data within ~~an~~ LDAP directory structure in a computer environment.

DESCRIPTION OF THE PRIOR ART

A Lightweight Directory Access Protocol (LDAP) directory (such as Netscape Communications Corporation's Directory Server) is a collection of "entries." Each entry has a name (called the Distinguished Name) and a list of attribute values. The entries in a directory are organized in a tree structure, with major groupings that are subdivided into smaller units. A directory might contain several organization entries, each of which contains several organizationalUnit entries. These entries can be further subdivided.

LDAP provides search operations that can be performed over specified portions of the directory tree. Trees and subtrees, therefore, are a natural way to deal with data stored in an LDAP directory.

Entries and attributes correspond to a wide variety of data types such as personnel information, server configuration, business relationships, and user preferences. Since all entries are stored within a single directory, a method is required to restrict the availability of specific information to authorized users.

The method used to control access is via Access Control Lists (ACL). The Directory Server Administrator (DSAdmin) creates some basic ACL rules that grant permission to

certain users to access various information in the directory. Most of the security considerations will require from tens to hundreds of rules to implement. The smaller number of ACL rules offers better performance and easier manageability.

Because a directory is the critical central repository in an intranet containing collections of information, *e.g.,* about people, it is imperative that a rich set of access options/features be provided. For example, the user should be able to modify his entry, or to update his home address or home phone number without any DSAdmin intervention.

A better feature would be to give the user the ability to decide who can access some of his personal information. The only way to do that is to allow users to create ACLs. However, a directory can contain millions of entries such as the directory used by Netscape's Netcenter. To support this size of a directory using the traditional approach would require millions of ACLs which would not only degrade the server's performance but would also be highly unmanageable. It also creates a risk, *i.e.,* the user can create a rule denying the DSAdmin some privileges which is unacceptable.

Another disadvantage is that ACL syntax are generally complex. A normal user is unable to understand the format and fields of the rules to be able to use the rules effectively and safely.

It would be advantageous to provide a simplified ~~LDAP~~ access control language ~~system~~ that gives the system administrator the ability to allow a user to specify a list of people that have access to certain attributes of that user's directory entry information. It would further be advantageous to provide a simplified ~~LDAP~~ access control language ~~system~~ that provides a simple mechanism to allow a user to make those specifications.

SUMMARY OF THE INVENTION

The invention provides a simplified LDAP access ~~language system~~. The system provides a simple command language that allows a system administrator to give a user the flexibility to specify a list of people that have access to certain attributes in a directory entry. In addition, the invention provides a ~~system~~ that allows a user to easily specify access lists without having to learn a complicated command syntax.

A preferred embodiment of the invention provides user-defined attributes that tell the directory system who the user wants to give read or write access to a specific set of his attributes. The read and write attributes are separate lists and may, in fact, differ. This gives the user the flexibility to better manage access to his attributes.

The value of the read and write attributes are in an LDAP Filter format which is an Internet standard (RFC 2254). The filter properties allow the user to specify not only users local to his intranet, but users across the Internet as well.

Access control lists (ACL) are created by the System Administrators. The ACLs list the specific attributes that the user is allowed to control read or write access. This gives the Administrators full control of what information the user can give out.

The ACLs are stored in the directory along with the entries. When a user accesses an entry in a directory, the server checks the ACL specified for the attributes being accessed. The read or write attribute for the owner of the attributes being accessed are used by the server when it checks the ACL. The combination of the read or write attribute and the ACL determine whether the user has permission to perform the read or write access to the attribute being accessed.

Other aspects and advantages of the invention will become apparent from the following detailed description in combination with the accompanying drawings, illustrating, by way of example, the principles of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a diagram of an LDAP directory entry according to the invention;

Fig. 2 is a schematic diagram of an example of how n attributes are accessed according to the invention;

Fig. 3 is a block schematic diagram depicting the organization of users in a company hierarchical tree according to the invention; and

Fig. 4 is a block schematic diagram of a directory including ACLs, entries and read/write attributes according to the invention.

DETAILED DESCRIPTION OF THE INVENTION

The invention is embodied in a simplified LDAP access control language system in a computer environment. A system according to the invention provides a simple command language that allows a system administrator to give a user the flexibility to specify a list of people that have access to certain attributes in a directory entry. In addition, the invention provides a system that allows a user to easily specify access lists without having to learn a complicated ~~command~~ syntax.

A Lightweight Directory Access Protocol (LDAP) directory (such as Netscape Communications Corporation's Directory Server) is a collection of "entries." Each entry has a name (called the Distinguished Name) and a list of attribute values. The entries in a directory are organized in a tree structure, with major groupings that are subdivided into smaller units. A directory might contain several organization entries, each of which contains several organizationalUnit entries. These entries can be further subdivided.

LDAP provides search operations that can be performed over specified portions of the directory tree. Trees and subtrees, therefore, are a natural way to deal with data stored in an LDAP directory.

Entries and attributes correspond to a wide variety of data types such as personnel information, server configuration, business relationships, and user preferences. Since all entries are stored within a single directory, a method is required to restrict the availability of specific information to authorized users.

The method used to control access is via Access Control Lists (ACL). The Directory Server Administrator (DSAdmin) creates some basic ACL rules that grant permission to certain users to access various information in the directory. Most of the security considerations will require from tens to hundreds of rules to implement. The smaller number of ACL rules offers better performance and easier manageability.

Because a directory is the critical central repository in an intranet containing collections of information, *e.g.*, about people, it is imperative that a rich set of access options/features be

provided. For example, the user should be able to modify his entry, or to update his home address or home phone number without any DSAdmin intervention.

A better feature would be to give the user the ability to decide who can access some of his personal information. The only way to do that is to allow users to create ACLs. However, a directory can contain millions of entries such as the directory used by Netscape's Netcenter. To support this size of a directory would require millions of ACLs which would not only degrade the server's performance but would also be highly unmanageable. It also creates a risk, *i.e.*, the user can create a rule denying the DSAdmin some privileges which is unacceptable.

Another disadvantage is that ACL syntax are generally complex. A normal user is unable to understand the format and fields of the rules to be able to use the rules effectively and safely.

The problems that are presented are:

- How to let users manage some of their own information.
- How can the DSAdmin manage the information so that no security rules are violated.
- How to make the server manageable under the above circumstances.

Ideally, a DSAdmin would like to have rules that perform the following tasks (out of n attributes in a directory):

1. Allow n1 attributes to be read by anyone in the world (this is a typical requirement). Example attributes are cn, sn, phonenumber.
2. Allow n2 attributes to be read and modifiable by the user himself, *e.g.*, home address, home phonenumber.
3. Allow n3 attributes to be managed by an owner/manager, *e.g.*, salary, employeegrade.
4. Allow n4 attributes to be managed by the user, *i.e.*, the user decides who can read or modify the attributes, For example, the user can decide that only Sam can read his hobbies attribute and only Kelly can read or change emergency contact info so that she can keep it up to date.
5. Do not allow the rest n5= [n - (n1 + n2 + n3 + n4)] attributes to be accessible to the general public except for the Administrator group, *e.g.*, employee status.

Solving 1, 2, 3 & 5 are fairly straightforward and ~~will be~~ explained below. The only difficult item is the 4th case. As previously mentioned, to enable this feature requires that the user be provided the ability to create his own ACLs. This could lead to millions of ACLs - which is not acceptable. The remainder of this document describes a novel approach to overcome these problems using a few ACLs and an existing Internet standard.

One area where this is applicable is in Netscape's Netcenter which has a registry of Netcenter members. All of the member information is stored in a directory server. The Netcenter Administrators would not only prefer to maintain the member's information/profiles but would also like to provide flexibility to allow members to maintain some other key information which other members can access. There is an immediate need with no current solution that can take care of this problem in an elegant way.

One skilled in the art will readily appreciate that although LDAP directories are mentioned throughout, the invention can be implemented in any directory application. Additionally, although the examples cited concern attributes pertaining to people, one skilled in the art will readily appreciate that the invention can control access to any attributes stored in a system.

Referring to Fig. 1, an example of a person's LDAP directory entry 101 is shown. Attributes are listed that pertain to a particular individual. Some of these attributes are controlled by the Administrator 102, *e.g.*, dn, sn, uid. A certain number of attributes 103 are the attributes that the user wants to control read and write access, *e.g.*, hobbies, homepage, personalpage.

The LDAP standard is very flexible. It allows extension of the schema by adding new attributes or objectclasses. One can add a new attribute called "hobbies" 107 to an entry as long as the objectclass which has that attribute has been added.

With respect to Fig. 2, the ideal situation is when, given n attributes 201, some of the attributes are public 202, *e.g.*, telephonenumber, where any user can see them. Other attributes are private and are not normally available to other users. For example, department admins and managers can only access a person's salary attribute 203 or only super admins can access a person's employee status 204. The final set of attributes are the ones that the user controls 205, *e.g.*, hobbies, homepage, and personalpage.

The following ACL syntax is used to explain how cases 1, 2, 3, & 5 are solved. Note, the syntax is used for reference only.

- Allow n1 attributes to be read by anyone:
  ACL: (list of n1 attrs) (allow (read) user = "anyone")
- Allow n2 attributes to be read/writeable by self:
  ACL: (list of n2 attrs) (allow (read, write) user = "self")
- Allow n5 attrs to be read/writeable by the admin group only:
  ACL: (list of n5 attrs) (allow (read, write) group = "Admingroup")
- Allow the owner or the manager to manage the n3 attributes:
  ACL:(list of n3 attributes) (allow (read, write) attr="manager" or attr = "owner")

Referring again to Fig. 1, user Prasanta's 108 manager is Claire 109 and she can read/write the list of n3 attributes. Similarly, Joe's manager, Bill, can read/modify Joe's n3 attributes. This is achieved by using one ACL. The value of the "manager" is plugged in at runtime.

For case 4, the requirements are more complex. Out of the n4 attributes, a finer granularity must be achieved, *i.e.,* the n4 attributes can be read by certain people (n4-read attrs) and can be modified by certain people (not necessarily the same people that can read the attributes) (n4-write attrs). A preferred embodiment of the invention solves the 4th case by providing an ACL similar to the other cases, but also using user-defined attributes.

Referring again to Fig. 1, the user-defined attributes 104 tell the system who the user wants to give permission to read 105 and write 106 to his attributes 103. It is evident that the read 105 and write 106 attributes are separate lists and may, in fact, differ. This gives the user the flexibility to better manage access to his attributes.

The invention's ACL syntax is as follows (using the example discussed earlier):

ACL: (list of n4-read attrs) (allow (read) filterattr= "whocanreadattr")
Ex: (hobbies, emergencyContact) (allow (read) filterattr= " whocanreadattr ")

ACL: (list of n4-write attrs) (allow (write) filterattr= "whocanwriteattr")
Ex: (emergencyContact) (allow (write) filterattr= " whocanwriteattr ")

where the values of a whocanreadattr & whocanwriteattr are:

whocanreadattr: (((ldap:///o=abc.com?(uid=sam)) (uid=kelly))

whocanwriteattr: (uid=kelly)

The value of the read and write attributes are in an LDAP Filter format which is an Internet standard (RFC 2254). The ACLs are created by the DSAdmin. This gives the DSAdmin full control of what information the user can give out. With respect to Fig. 3, the filter properties allow the user to specify not only users local to his intranet, but users across the Internet as well. In the example above, Prasanta 303 and Kelly 304 are users 302 in the same company, Netscape 301. The "whocanreadattr" gives Sam who is from ABC company and Kelly who is from Netscape, read access.

Referring to Fig. 4, the ACLs 402 are stored in the directory 401 along with the entries 403. When a user accesses an entry 403 in a directory 401, the server checks the ACL 402 specified for the attributes being accessed. The read 404 or write 405 attribute for the owner of the attributes being accessed are used by the server when it checks the ACL 402.

Using the above example, the value of "whocanwriteattr" is plugged in by the server at runtime with "(uid=kelly)". So, if Kelly is the accessing client, the filter matches to TRUE and Kelly is allowed to modify the "emergencyContact" attribute. However, if Bill is the client, the filter matches to FALSE and Bill is denied the privilege. Each user can now create LDAP Filters which will allow them to manage their own information.

The advantages of the invention are:

- The Admin has complete control of what a user can do.
- Only a handful of ACLs are needed instead of millions.
- The performance of the server is markedly increased.
- The value of the new attributes are based on an Internet standard.

Although the invention is described herein with reference to the preferred embodiment, one skilled in the art will readily appreciate that other applications may be substituted for those

set forth herein without departing from the spirit and scope of the present invention. Accordingly, the invention should only be limited by the Claims included below.